技术变化太快，程序员咋办？ 从Adobe Flash想到那些年我幸运躲过的MFC和塞班

Original 刘超 刘超的通俗云计算 *2017-07-31*

随着年龄越来越大，越来越开始觉得技术变化快给程序员的压力也越来越大。

这篇文章是我的职业生涯的历程，和一些思考。

2005年的时候，MFC当时还是非常火的，当时互联网应用还没有那么兴起，手机还是老版诺基亚的手机，所以桌面应用的开发还是有一席之地的。

当时连大学里面的老师都不怎么教Linux下的程序设计，C++程序设计课统统用的是Virtual C++，网络实验课也是用Virtual C++做了一个TCP互联进行聊天的程序，能打开打印机已经是非常牛的事情了。

于是当时本来下决心要把MFC学好，向往什么时候自己也能做一个Office之类的桌面程序，于是有一本大部头的书《深入浅出MFC》，侯杰写的，这是读的第一本关于源码分析的书，也立志自己将来也要对一个框架了解到如此的透彻，也能写本源码分析的书。

MFC做了不到一年，发现了不爽的方面，就是无论你对这个框架如何熟，到了最底层能接触到的也只有微软写的API，而再往下就只能看MSDN中抽象的解释了，代码？嘿嘿，微软不会给你看的，就算有源码出来，也是过时的，而且没有像Linux一个社区一样很多人去解析，一大堆代码放在那里，全无头绪。

而且当时在思考一个终极问题，也是很多非计算机专业的同学问的，你看你们学计算机的，网站做不了，桌面应用做不了，你们都学了些啥？我说我学得是计算机科学，例如操作系统，计算机网络，数据结构，数据库，编译原理这些核心课程啊，又不是做网站和桌面应用的。同学就问了，那你学这些东西也没啥用啊？这个问题我当时回答不了，但是我始终相信，这么多前辈科学家们总结出来的这些基础学科肯定有用处，是我的方向没选对，我不应该仅仅做一个拖拖拽拽就搞定的程序员。

于是我自己开始看APUE《Advanced Programming in the UNIX》，看《TCP/IP illustrated》，看《The C++ programming languages》，看《Understanding the linux kernel》等，我要找一个做Linux的工作，能用的上大学学得基础学科。这些都是大部头的书啊，一本比一本厚，但是这些书是都值得反复读的，哪怕从现在来看也是的。

很幸运从事MFC不到一年，就面上了当时还如日中天的诺基亚2006年，当时实习培训的时候，诺基亚的市场占有率94%，给我们培训的讲师说在手机市场，诺基亚说什么就是什么，当时没有任何怀疑。当时诺基亚的福利好的出乎意料，每周都有近郊团建，每年有两次出国。但是我还是感觉出来了不对的地方，当时在诺基亚加入的组是做后端开发的，应该是跑在类似基站里面的一个操作系统，整个操作系统全部都是诺基亚自己开发的，连程序设计语言都是诺基亚自己的，这个操作系统的唯一目标是稳定，因为一旦基站部署在山上，是没有人去维护的。不但是操作系统，就连整个项目管理系统，Bug系统，持续集成系统，自动化测试系统等，全部是诺基亚自己开发的，我当时想除非我要在诺基亚干一辈子，不然在这里待了五年，就只会这里面的东西了。不行，我要去一个用通用技术的地方，这是第一次，我爱上了开源的系统，应用开源，不但厂商不被绑定，程序员也不被绑定。

很幸运，由于我看了很多上面列出的书籍，成功面试上了EMC，当时EMC刚来中国，办公室还在上海静安寺，2016年7月14日，作为EMC中国研发中心的第一批员工，我入职了，并且很幸运的，我从事的是Linux相关的开发，并且开发的是一套分布式存储系统，从应用层到内核，到计算机网络，都用上了，第一次感觉到，大学学的东西真有用，而且在大学里还没学好。能和国内和国外的存储界大牛一起学习，真的很受益，这里面很多大牛的高瞻远瞩当时根本感觉不到，例如EMC的一个distinguished engineer，也就是仅次于fellow的一个大牛Fred Rivera，给我们说对象存储将来会大放异彩，当时AWS的S3推出不到三个月，当然现在大家都知道S3几乎成为大容量存储的一个业界标准。在当时看了AWS应该是主流做电商的，BAT也没有一个上市的，外企还风风火火，后来在五角场Baidu在EMC旁边，当时在Baidu上市之前，还有很多跳到EMC的，真是肠子都悔青了，但是后来看多了，也发现财富这个事情听天由命，因为后来2010年的时候，碰到了一个当时在阿里就是P9的员工跳出来的，全是命。

还是接着说技术吧。当时有一个做销售的朋友，问我愿不愿意做手机端的开发，因为后来诺基亚有了智能机，就是上面是密密麻麻全键盘的那种，里面是塞班系统。销售的朋友说，他觉得手机移动应用将来是趋势，我应该率先行动起来，学移动应用开发。我说好，我业余研究一下，可是研究什么呢？没有别的选择，只有塞班，当时以诺基亚的实力，塞班绝对是标准。其实现在看来我的销售朋友的眼光很不错，趋势看的也对，但是如果我真的转了塞班，对于技术人员来讲，却是个灾难。做技术的千万不要相信什么精通了一门语言，所有的语言都精通，精通了一部分，其他的都触类旁通这种鬼话，你做了10年Java，去面试一个C++的职位你就知道了，而且越往后越难转。

当时我之所以没有学成塞班，是因为文档不全，找不到很好的示例，这也是为什么我喜欢开源的原因，官方文档往往新手上手不容易，而开源社区会有大量的文章和人帮你。而且后来在EMC，我进入了一个更有意思的项目组，就是搜索引擎，当时是2008年，我们基于的是开源框架Lucene。

搜索引擎太有意思了，而且当时我们做的是一个能够有事务支持的搜索引擎，要和一个数据库联动的。在这里面我用到了编译原理，因为解析一个用户查询需要解析语法树，能用的JavaCC，用到了多种数据结构，Trie树，B+树，倒排表，跳跃表等，用到了数据库的write ahead log实现CRUD，因为自己要关心事务的事情。

而且最最重要的，这是我接触过的第一个开源软件，并可以通过阅读开源代码了解背后的机制，并能够进行定制化。而且这燃起了我想成为另一个侯杰的梦想，我要写博客，我要分析源代码，分析的越透彻越好，如果能够写一本书就好了。

于是我在CSDN和JavaEye，现在合并成一家，后者叫ITEye了，注册了一个叫做forfuture1978的账号，名字叫觉先，应该做Lucene的可能听说过这个名字。我不是78年的，是82年的，当时为了显老，显得权威一些，所以用了这个名字。之所以叫觉先，是因为当时看了吴晓波的《激荡三十年》，有一个叫陈春先的人，这个人本来是一个和陈景润齐名的科学家，可是从美国硅谷回来之后，立志要创办中国的硅谷中关村，于是自己创业了，没有在科学家的道路上走下去，被称为中关村民营科技第一人，当然作为科学家的他商业头脑不灵，没有像柳传志一样赚大钱，却牺牲了自己的前途，作为IT的从业人员，看到这里很感动，于是记错了名字，记成立陈觉先，于是1978年是改革开放元年，于是我起名字博客为觉先。

当时开始对Lucene的源代码进行深入的分析，后来形成了《Lucene原理与代码分析》http://download.csdn.net/detail/forfuture1978/2452992，长达500多页，霸占JavaEye下载第一名好久，里面恨不得每一行代码都解析过了。当然这本书起到的第二个作用是机械工业出版社和华章培训网邀请出了一个课程，《Lucene应用开发解密》

https://baike.baidu.com/item/Lucene%E5%BA%94%E7%94%A8%E5%BC%80%E5%8F%91%E6%8F%AD%E7%A7%98/10540077?fr=aladdin

但是做Lucene的代码分析的时候，当时的一个误区是太关注代码的细节，每一行都要扣懂了才罢，然而开源软件更新实在太快了，技术的迭代太快了，根本赶不上。所以后来在开源软件分析方面，总结出来的经验是，代码是要看，但是要超脱出来看思想，或者再深入一层看原理，而不纠结与代码细节，思想和原理是相对稳定的，这一点收益良多。

于是在公众号里面，有关搜索引擎原理的，有下面的几篇
不是技术也能看懂搜索引擎
大数据就是众人拾柴火焰高

有关技术本质的又下面的几篇
搜索引擎的设计(1)：词典的设计
搜索引擎的设计(2)：倒排表的设计上
搜索引擎的设计(3)：倒排表的设计下

这里面的数据结构，压缩算法，是至今不做搜索引擎都有用的，例如在前几天AWS的submmit上面听到的在对象存储中应用到的压缩算法，有了这个基础，就一听就能听明白。

当Lucene研究的比较深入之后，在2009年出来加入了一家创业公司，专门做搜索引擎。当时搜索引擎太热了，因为Google刚退出中国，连乒乓球运动员都来做搜索引擎，于是跟随了两个技术大拿一起创业了，一个是美国的海归，放弃了Linkedin的上市机会，一个是阿里的P9，放弃了阿里的上市机会。现在那家公司依然还活的非常好，两个老大都还在。

跟着这两个老大学到的一点，就是创业的心态和开放的心态。老大说创业的工作就是要学会Drive，如果觉得某件事情能够促使组织进步，哪怕不是你的工作范围内的，你也要驱动自己或者他人去做，要有开放的心态，碰到新的技术不要抵触，有用就拿来用，不要抱着自己的一亩三分地的技术不放。这点的确有用，因为技术变的太快的，你自己抱着，觉得自己的技术有用，别人的技术不行，时过境迁，三十年河东三十年河西，有多少抱着自己的技术不教新人的老员工同时自己也不进步，从而被淘汰。

在这家创业公司，第一次接触了SOA，并且接触到了一个从搜狐过来的架构师搭建的一套服务发现的系统，注册中心用的不是zookeeper，而是数据库，但无所谓，原理都是一样的，这给我后面接触微服务和Dubbo，SpringCloud奠定了很好的基础。旁边坐着一个hadoop的大牛，在这里面学习了map-reduce的原理，也看了源代码，虽然后来map-reduce有了2.0，后来又有了spark，但是从原理来讲，分治法处理的思想是不变的，能够比较容易上手。

Hadoop源码分析(1)：HDFS读写过程解析

http://www.cnblogs.com/popsuper1982/p/5585411.html

hadoop源码分析(2)：Map-Reduce的过程解析

http://www.cnblogs.com/popsuper1982/p/5585420.html

Hadoop源码分析(3)：Hadoop的运行痕迹

http://www.cnblogs.com/popsuper1982/p/5585424.html

Hadoop源码分析：Hadoop编程思想

http://www.cnblogs.com/popsuper1982/p/5585427.html

这是四篇hadoop的原理与代码分析的文章，没有分析的很细，而且是map-reduce 1.0版本的，所以没有放到公众号里面。

到了这个时间，年级大了，于是重新使用了popsuper1982这个博客。

另外在这家创业公司，还接触到了一些对于财经领域的分类和聚类的算法，以及NLP的算法，当时还没有深度学习，主流的方式还是基于概率的。也是这一段经历，使得现在接触深度学习的时候，有了一定的基础，不是完全从头开始。

通过这家创业公司，我也发现，对于一个技术人员来讲，大公司是做深入的好机会，而创业公司是做广的好机会，技术人员眼光还是要开放，因为可能接触到的一些点，以后会有大的用处。

当然接触Flash Air的开发也是在这家创业公司，当时Flash Air如日中天，所以我们的财经分析软件就是用Air开发的，很酷。但是这个技术我不喜欢，是因为被一家公司主导，不开源，当时也没想到Flash会被Adobe放弃掉。

后来来到了HP做云计算，当时还没有什么OpenStack啥的，Vmware是私有云老大，AWS当时已经是公有云老大，最初是去HP做云管平台的。为什么能够面试上HP呢，是因为云管平台要管理hadoop的平台，这是结合点，那为什么我要来HP呢？是因为这是一个Solution Archtect的职位，能够接触客户，因为上一家创业公司的两个老大都是技术出身，思维都太技术了，我当时太想知道用户想什么了。

但是我又不想做一个完全念PPT的工作，于是还是参与了一些运维和实施的工作，好在HP比较开放，没有太严格的界限。

在学云计算的时候，我继续秉承了了解思想的方法，通过了解Vmware和AWS，了解作为云计算，从设计角度应该有哪些功能。然而想深入了解原理，很可惜Vmware和AWS是黑盒的，光看文档不解渴，好在后来有了OpenStack，HP有了Helion。

OpenStack什么都在学AWS，所以我开始看OpenStack的代码，但是着重关注其中的原理部分，而不必过分在意OpenStack的代码的细节。于是就有了《OpenStack创建虚拟机的50个步骤和100个知识点》。

OpenStack虚拟机创建的50个步骤和100个知识点

50个步骤，可能随着后来代码的改变，早就不是这50个步骤了，但是100个知识点，从认证，鉴权，调度，高可用，虚拟化，网络，存储等都是不变的。

我可以不跟进OpenStack的代码变化，但是我要了解KVM的细节，这就是为什么公众号有一大批是关于KVM的。

我是虚拟机内核我困惑？！
Qemu，KVM，Virsh傻傻的分不清
裸用KVM创建虚拟机，体验virtualbox为你做的10件事情
用OpenStack界面轻松创建虚拟机的你，看得懂虚拟机启动的这24个参数么？
手动用KVM模拟OpenStack Cinder挂载iSCSI卷
KVM半虚拟化设备virtio及性能调优最佳实践
我的虚拟机挂了！怎么把镜像里面的数据找回来？

我也要了解网络的细节，于是公众号有一大批是关于OpenVswitch的。
觉得OpenStack的网络复杂？其实你家里就有同样一个网络
当发现你的OpenStack虚拟机网络有问题，不妨先试一下这16个步骤
Openvswitch总体架构与代码结构
跟随Openvswitch数据库，才能玩遍OVS
玩转Openvwitch第一站：Manager和SSL
玩转Openvswitch第十站：Flow Table
玩转Openvswitch之综合篇
从Openvswitch代码看网络包的旅程

接下来其他部分也会逐渐写进去。

看懂了OpenStack，再回来看AWS，则理解的更加深入了，有点像穿着透明的衣服，你大概知道AWS的这个界面操作后台是怎么弄的。

但是HP的Helion OpenStack在国内水土不服，于是后来就到了华为，从事FusionSphere OpenStack相关的工作，也接触了不少的客户，不但了解了云计算技术，对于云计算的产业，竞争对手的情况，各个厂家的优劣势的比较，也做到心中有数，有了一定的大局观。

看懂了OpenStack，后来就有了Docker，并且加入了一家做容器的创业公司。OpenStack还没火完，Docker又来了，技术更新也太快了，但是当你掌握了原理，就发现其实没变什么，认证和鉴权，AWS是这样做的，OpenStack是这样做到，Mesos是这样做到，Kubernetes也是这样做的，Namespace和cgroup，会Linux的都应该会的，会了网络的基本原理，什么Docker Overlay，Weave，Flannel，Calico等等不一而足，但是往细里看，还是那些基本原理。

当然有好的风景也不能错过，心态要开放嘛，比如Mesos的两层调度，原来没有接触过，就需要仔细研究一下。
号称了解mesos双层调度的你，先来回答下面这五个问题！

在容器创业公司，又是我一个广度拓展的过程，Swarm，Mesos，Kubernetes都用上了，还接触到了NFV，4G网络的架构，E-NodeB，MME，SGW，PGW，HSS，PCRF等，以及管理面的Diameter协议，数据面的GTP协议，有空可以写

写这部分，不知道以后这部分知识是否能够用到。还接触到了SMACK，Spark Mesos AKKA Cassandra Kafka。而且部署的应用是springboot + dubbo的，在SOA方面又进一步。

后来来到了网易云，从事解决方案总架构师的职位。在网易，kubernetes需要更加深入的研究一下，因为难得能够碰到有如此大规模的kubernetes集群，微服务也需要深入研究一下，因为难得能够见证考拉和云音乐如此迅猛的增长。另外还会继续保持开放的心态，对大数据和人工智能部分做一些了解，能够如此早的将人工智能用于类似客服和业务安全的厂商也不多，大部分还处于预研究的状态。

接下来说说我的感悟吧：

- 心态要开放，做程序员，就做老罗说的终身学习者吧，别抱着一个技术不放，害怕别人学会，因为很快会过时，别对新产生的技术抱敌对的态度，这没有任何用处。
- 学习什么呢？要学那些开源的，至少开放的技术，而且越主流，热度越高越好，不要相信单个企业的强大，另一个罗胖也说过，现在牛的冒烟的企业，估计30多年，也就差不多了。要相信生态的力量，蚂蚁见证了恐龙的诞生，也见证了恐龙的灭亡，见证了人类的诞生，估计也会见证人类的灭亡。那些已经被广泛使用的开源软件就像蚂蚁。
- 学习开源软件不要纠结于代码层面，而是向下深入挖掘本质的原理，这些原理正是大学里学的这些课程，是科学家们智慧的结晶，就像分子一样，就算有一天蚂蚁灭绝了，也不过是分子的重新组合。
- 学习开源软件还可以向上抽象出思想，这些思想都是可以用于产品设计和架构设计的。
- 懂一些市场，懂一些客户，从市场和客户那里是能够发现技术的发展方向。